EVENTs Socket Documentation

1. Join Room

Event: `join_room`

Description: This event is triggered when a user opens a specific Conversation.

Parameters:

- `data` (String): The room ID.
- `userId` (String): The ID of the user joining the room.
- `unseenMsg` (Array): An array of unseen message IDs for the user in the room.

Usage:

```
    socket.emit("join_room", roomId, userId, unseenMsg);
```

Response: Emits `all-msg` event with all messages in the room.

Event Listener on frontend:

```
 socket.on("all-msg", (messages) => {
  // Handle received messages
});
```

---

2. Leave Room

Event: `leave_room`

Description: This event is triggered when a user closes a specific Conversation.

Parameters:

- `data` (String): The room ID.

Usage:

```
socket.emit("leave_room", roomId);
```

Response: no response

---

3. User Conversations

Event: `all_conv`

Description: This event is triggered to fetch all conversations for a specific user.

Parameters:

- `userId` (String): The ID of the user.

Usage:

```
socket.emit("all_conv", userId);
```

Response: Emits `all_conv` event with the user's conversations.

Event Listener on frontend:

```
socket.on("all_conv", (conversations) => {
  // Handle received conversations
});
```

---

4. Send Message

Event: `send_message`

Description: This event is triggered when a user sends a message.

Parameters:

- `data` (Object): An object containing message details.
  - `room` (String): The room ID.
  - `sender` (String): The sender's ID.
  - `text` (String): The message text.
- `senderName` (String): The name of the sender.
- `nextUserId` (String): The ID of the recipient user.

Usage:

```
socket.emit("send_message", {
  room: roomId,
  sender: senderId,
  text: messageText
}, senderName, nextUserId);
```

Response: Emits `receive_message` event with the saved message and room members if the recipient is online, or `count` event if the recipient is not in the room.

Event Listener on frontend:

```
socket.on("receive_message", (message, roomMembers) => {
  // Handle received message
});
socket.on("count", (conversationId) => {
  // Handle unseen message count
});
```

5. Seen Message

Event: `seen_msg`

Description: This event is triggered when a user has seen a message.

Parameters:

- `msg` (Object): The message object.
- `userId` (String): The ID of the user who has seen the message.

Usage:

```
socket.emit("seen_msg", message, userId);
```

Response: No response
